Massachusetts Institute of Technology
Instrumentation Laboratory
Cambridge, Massachusetts

AGC Programming Memo #25

TO: Distribution

FROM: D. Eyles

DATE: May 3, 1967

SUBJECT: New GEOMETRY Routines

This memorandum introduces to AGC programmers the routines that have replaced SMNB, NBSM, and AXISROT in the GEOMETRY subroutine.

I. HISTORY

Since the vector transformation routines SMNB and NBSM (which share the subroutine AXISROT) are in some guidance phases called twice every guidance period (by the find CDU desired program), an effort to reduce their excessive execution time was thought worthwhile. It was believed that a basic language version could be written that would require little, if any, more memory and would improve execution time by a factor of at least two -- without, naturally, sacrificing accuracy. AX*SR*T, and its ancillaries, was the result.

II. DESCRIPTION

AX*SR*T combines the old SMNB, NBSM, and AXISROT. It is a strict subroutine, called using a TC. User signals which transformation he desires using the contents of A. +3 indicates the nav base (NB) to stable member (SM) transformation. -3 signals SM to NB. Thus the calling sequences are, for NBSM:

```
CA      THREE
TC      AX*SR*T
- return here -
```

and for SMNB:

```
CS      THREE
TC      AX*SR*T
- return here -
```

Most users, however, will prefer to use one of the several interface routines that are provided (see below for details).

The vector to be transformed arrives, and is returned, in the 6 locations beginning at VBUF. Mathematically AX*SR*T, like AXISROT, performs the transformation by rotating the vector in turn through the three Euler angles relating the two coordinate systems. This was found to be easier and quicker than multiplication by a matrix. AX*SR*t expects to find the sines and cosines of the angles of rotation -- in general CDU angles -- at SINCDU and COSCDU, in the order Y Z X. A call to CD*TR*GS (see below) takes care of this.

AX*SR*T is no less accurate than the extinct routines. It is guaranteed safe (only) for vectors of magnitude less than unity. A look at the case in which a vector of greater magnitude happens to lie entirely along an axis of the system to which it is to be transformed convinces one that this is a restriction which must be accepted.

CD*TR*GS computes the sines and cosines of the 2's complement angles it finds at CDUSPOT and stores the results at SINCDU and COSCDU. For CD*TR*GS the angles should appear, each single precision, at CDUSPOT, CDUSPOT +2, and CDUSPOT +4; odd locations need not be zeroed. The sines and cosines are placed in SINCDU and COSCDU in the same order as the input angles. Thus, if CD*TR*GS is being called as preparation for AX*SR*T the angles must have been placed at CDUSPOT in the order Y Z X. (Users may find the RTB op code READCDUS helpful in this respect.) Note that CD*TR*GS destroys part of VBUF; thus if CD*TR*GS and AX*SR*T are being called in succession the vector must be placed at VBUF <u>after</u> the call to CD*TR*GS. Except for the generally insignificant MPAC +2, CD*TR*GS leaves the MPAC area as it finds it -- incidently.

AX*SR*T does not destroy the values at SINCDU and COSCDU. Thus the call to CD*TR*GS need not be repeated, when AX*SR*T is called more than once, unless the angles have changed. This, and the fact that the sines and cosines remain available to the user, can be the source of significant time savings.

Four permanent interface routines are provided. All restore user's EBANK setting. All are called from interpretive using "CALL" and return via QPRET. All expect and return the vector to be transformed interpreter-style at MPAC; components at MPAC, MPAC +3, and MPAC +5.

TRG*NBSM and TRG*SMNB both expect to see the 2's complement angles at CDUSPOT (order Y Z X). TRG*NBSM does the NB to SM transformation; TRG*SMNB, vice versa.

NBSM and *SMNB* expect to see the sines and cosines rather than the angles themselves. Otherwise they are like, respectively, TRG*NBSM and TRG*SMNB.

Note that just as CD*TR*GS need be called only once for each series of transformations using the same angles, so too only one of TRG*SMNB and TRG*NBSM need be called for each series.

Summary of execution times (all are approximate and are in the presence of interrupts):

AX*SR*T	13 ms.	
CD*TR*GS	54 ms.	
TRG*SMNB	62 ms.	
TRG*NBSM	62 ms.	
SMNB	14 ms.	
NBSM	14 ms.	
SMNB	119 ms.	extinct routines
NBSM	124 ms.	extinct routines

III. EXHORTATION

Time savings, as illustrated above, were effected by the use of basic language; memory savings by machiavellian means. Rewriting interpretive coding in the vernacular was found to be FUN, and, if only as an exercise, is recommended.